REMARKS

Applicants wish to thank the Examiner for the courtesies extended during the telephone interview held on March 19, 2008. Applicants believe that the amendments and remarks herein are consistent with the substance of that interview.

The most recent Office Action mailed July 12, 2007 ("*Office Action*") considered claims 1-27. The *Office Action* rejected claims 1-6, 9, 12-15, 20-21, and 25-26 under 35 U.S.C. § 103(a) as being anticipated by S. Eisenbach et al., *Managing the Evolution of .NET Programs*, FMOODS 2003, LNCS 2884, pp. 185-198, 2003 ("*Eisenbach*") in view of M. Gunderloy, *Managing Versions of an Application*, Feb. 2002, Lark Group, Inc., pp. 1-6 ("*Gunderloy*"). The *Office Action* also rejected claims 7-8, 10-11, and 16-19, as being unpatentable over *Gunderloy* in view of S. Pratschner, *Simplifying Deployment and Solving DLL Hell with the .NET Framework*, Nov. 2001, Microsoft Corporation, pp. 1-12 ("*Pratschner*").[1] In addition, the *Office Action* rejected claims 22-24 and 27 under 35 U.S.C. § 102(a) as being anticipated by *Eisenbach*.

With this paper, Applicants have amended claims 1, 4, 6-7, 10, 16, 20-23, and 26-27, and have further cancelled claims 2-3, and 24-25. Accordingly, claims 1, 4-23, and 26-27 are currently pending, of which claims 1, 20, 22, and 26-27 are independent. Of these, claim 20 (and 21) is a functional claim corresponding to the limitations of claim 1. In addition, claims 22 and 27 are computer program product claims corresponding to the limitations of claims 1 and 26, respectively.

[1] Applicants reserve the right to challenge the sufficiency of the *Gunderloy* and *Pratschner* references under 102(b) as there may be some question as to the publication date(s), as these documents appear to be only online publications.

Applicants' invention as generally recited in amended claims 1, 20, and/or 22 relates to a system that automatically and differentially provides target component access to a requesting component, such as based on whether the target components are platform or library components. For example, and as described throughout Applicants' specification, target components that qualify as "platform components" can be provided to requesting components on a dynamic basis based on whatever happens to be at least the earliest version of the platform component that the requesting component can accept. *E.g.*, ¶ 32 of Applicant's Application Publication. Platform components can also be overwritten by new "versions" of platform components. *Id.* By contrast, "library components" are those components that are typically or rarely overwritten, and generally maintained side-by-side with other components in the system. *Id.* The system thus provides access to various versions of library components when the requesting component asks for a specific version of that target component. *Id.*; *see also* ¶¶ 36, 41-42. This differential, hybrid approach to maintaining and providing target components on the system as taught and claimed by Applicants provides a developer and/or administrator with a much greater deal of flexibility and stability in component-component access requests, without necessarily mandating the all-or-nothing approaches found in the cited art.

For example, the cited *Eisenbach* reference discloses a mechanism for overcoming the challenges of "dynamic linking" in systems in which newer and older versions of target components are maintained side-by-side. *Compare*, pg. 185, ¶1, *with* pg. 185, ¶ 3 – pg. 186., ¶ 4. Dynamic linking, in turn, refers to links between requesting components and target components that are determined and/or implemented only at runtime. As *Eisenbach's* disclosed mechanisms appear to be based primarily on a system in which older versions of components are

not overwritten or removed, *Eisenbach* fails to disclose – and moreover teaches away from – a system that employs a hybrid, differential approach, such as taught and claimed by Applicants, where some components can be overwritten or removed. This is not surprising since *Eisenbach* describes the overwriting of older components as one of the bases contributing to what is commonly known as "DLL Hell." *See* pg. 185, ¶ 2.

Specifically, Applicants can find nothing in *Eisenbach*, whether singly, or in combination with the *Gunderloy* or *Pratschner* references, which teaches, suggests, or describes that access to one or more target components can be provided "on a differential basis from one target component to the next" where "platform components" are provided based on the latest acceptable version, and "library components" are provided only per the requested, specified version. Similarly, *Gunderloy* fails to supply these limitations, particularly with respect to the automated nature of these determinations.

For example, and as previously described in Applicants' last response, *Gunderloy* teaches that a system provides a "default" version to requesting components, which can only be overridden by "explicit" user input into a particular interface for a given component. The *Gunderloy* reference indicates that when the user (*e.g.*, application developer) creates (*i.e.*, "you create") a component, the user can fill out various parts of a version number for a given component. *Gunderloy*, pp. 1-3. The user can also indicate the specific version of a target component that the requesting component will need. *Id.* If the user does not "explicitly" request a specific version number, however, the system in *Gunderloy* provides a requesting component with a "default" version of the target component, which is the version of a target component available when the requesting component was built/installed. *Id.*; *see also*, pg. 3 (stating, "you'll

see that even though there is a more recent version of the library installed, the older version will be used by the client application until you explicitly construct an application configuration file.") In some cases, *Gunderloy* indicates that the system may even miss any user input (and thus provide a default version) if the user fails to create a policy file using "strong names."

Accordingly, Applicants respectfully submit that both *Eisenbach* and *Gunderloy* (as well as *Pratschner*) fail to disclose a system that implements both library and platform components, and further automatically determines how to provide such components to requesting components based on versioning policies. Applicants respectfully submit, therefore, that claims 1, 20-21, and 26 (and the corresponding dependent claims) are allowable for at least these reasons.

For similar reasons, Applicants also respectfully submit that claims 22 and 27 are allowable over *Eisenbach*, whether singly, or in combination with any of the *Gunderloy* and/or *Pratschner* references. For example, claims 22 and 27 disclose a system that automatically determines how to handle the installation of different versions of various target components on a differential basis, based on an identification through the corresponding versioning policy of whether the target component is a library component or a target component. At the outset, Applicants respectfully submit that none of the references of record specifically deals with mechanisms for "automatically managing access of one or more versions of computer-executable target components." As previously mentioned, for example, *Eisenbach* deals explicitly with a system that manages dynamic linking between components at runtime, where all versions of the target components are all maintained side-by-side in the system. In addition, *Gunderloy* discloses a system for managing links between components, which requires explicit user involvement in the component designations.

By contrast, Applicants can find nothing in *Eisenbach*, *Gunderloy*, and/or *Pratschner*, whether singly or in combination, which teaches, discloses, or even suggests that a system automatically determines whether to delete earlier versions of a platform component, or simply maintain all versions of a library component on a system, based on the designation of library component or platform component in the corresponding versioning policies. Applicants respectfully submit, therefore, that independent claims 22 and 27 (and the corresponding dependent claims) are also allowable for at least these reasons.

In addition to the foregoing, Applicants have made a number of amendments to the dependent claims to clarify one or more features of Applicants' claims. Although not specifically addressed herein, Applicants respectfully submit that each of these amendments are also allowable over the references of record.

In view of the foregoing, Applicants respectfully submit that the other rejections to the claims are now moot and do not, therefore, need to be addressed individually at this time. It will be appreciated, however, that this should not be construed as Applicants acquiescing to any of the purported teachings or assertions made in the last action regarding the cited art or the pending application, including any official notice. Instead, Applicants reserve the right to challenge any of the purported teachings or assertions made in the last action at any appropriate time in the future, should the need arise. Furthermore, to the extent that the Examiner has relied on any Official Notice, explicitly or implicitly, Applicants specifically request that the Examiner provide references supporting the teachings officially noticed, as well as the required motivation or suggestion to combine the relied upon notice with the other art of record.

In the event that the Examiner finds remaining impediment to a prompt allowance of this application that may be clarified through a telephone interview, the Examiner is requested to contact the undersigned attorney.

Dated this 20th day of March, 2008.

Respectfully submitted,

/Michael J. Frodsham/

RICK D. NYDEGGER
Registration No. 28,651
MICHAEL J. FRODSHAM
Registration No. 48,699

Attorneys for Applicant
Customer No. 047973

MJF:kbl
W:\13768\328\KJB0000008428V001.DOC